Reconfigurable Computing
Technology Roadmap

October 2001

Submitted to the Earth Science Technology Office (ESTO)
under contract NAS5-99001 Task 4 by:

PARADIGM COMPUTING CORPORATION
a Limited Liability Company
1906 R Street N.W. Suite 1
Washington, DC 20009
Phone: 202.299.0917
Fax: 202.299.0918
configware@paradigm-computing.com

Table of Contents

Summary

As a paradigm nears its point of exhaustion, a new one, much more powerful and complex, starts its evolution. The new emergent cycle is covered by the current and successful paradigm, and clouded by its own state of disorder. As the new paradigm is consolidated, it suddenly appears from nowhere. It is very common that organizations and entire industries fail to recognize the formation of a new paradigm, and as a result do not participate in its creation.

There is enough evidence that shows the existence of a paradigm shift in computer design. For about half a century, the software paradigm, based on the Von Neumann compute engine, has been the major force in the development of computer technology. The advent of the Field Programmable Gate Array about 15 years ago started a new cycle with the exploration of reconfigurable computing techniques. Today, reconfigurable computing technology enables the emergence of the configware paradigm. The configware paradigm is orders of magnitude more complex and powerful than the software paradigm.

The configware paradigm will enable NASA to pursue scientific missions not possible today. It is required that NASA take a proactive role in steering the research community towards solid milestones for the continuing development of reconfigurable computing technology. Much of the research done to date follows a track of short-term compensation. A responsible, long-term approach, realizes that it could take more than a decade for the new paradigm to emerge. This understanding would best benefit the actions that can truly contribute to the creation of the configware paradigm.

Reconfigurable computing already enables a more advanced spacecraft design, one that shows computing power similar to that found on ground systems today. However, breaking away from the current mindset requires more than a traditional technology development and infusion approach. It requires a managerial commitment to a long-term plan to explore new thinking in computer science and engineering.

Introduction

The NASA Strategic Plan 2000 establishes as one of NASA's fundamental questions the ability to harness revolutionary technological advances to provide air and space travel for anyone, anytime, anywhere, more safely, and more affordably. It also questions which cutting edge technologies, processes, techniques, and engineering capabilities must be developed to advance NASA's research agenda in the most productive, safe, economical, and timely manner [1,4].

The Earth Science Enterprise (ESE) technology goals are to develop advanced technologies to reduce the cost and expand the capability for scientific Earth observation, develop advanced information technologies for processing, archiving, accessing, visualizing, and communicating Earth science data, and to partner with other agencies to develop and implement better methods for using remotely sensed observations in Earth system monitoring and prediction [2].

The ESE has identified among its near term goals (2000 to 2005) the need to employ high-performance computing to address Earth system modeling challenges, validate revolutionary technologies and satellite formation flying, and explore new instrument concepts. Its mid-term goals (2006 to 2011) include the infusion of revolutionary technologies into operational missions, the employment of distributed computing for Earth system modeling, the development of autonomous satellite control and advanced instruments, the implementation of a new generation of small instruments, and the development of onboard data processing and storage. For its long-term goals (2012 to 2025), the Earth Science Enterprise intends to deploy cooperative satellite constellations, enable intelligent sensor webs, develop advanced instruments for observations from liberation points, create sub-miniature spacecraft, and to develop new technologies that allow NASA to conduct missions with lower mass and power.

Reconfigurable computing, the use of Field Programmable Gate Arrays (FPGA) for data processing, has been identified as one of the emerging technologies that promises a breakthrough advancement in our ability to process data at faster speeds and lower cost [5, 6, 8]. The ESE investment in the development of reconfigurable computing technology is well aligned with its near, mid and long term goals. The advancement of reconfigurable computing enables the development of higher-performance and lower-cost computing systems for both ground and spacecraft data processing systems. For example, the concept of a single-chip spacecraft is feasible utilizing a high-density FPGA. The single-FPGA spacecraft is totally reconfigurable and adaptive to diverse mission configurations such as satellite constellations, intelligent sensor webs, and sub-miniature spacecraft. Conversely, high-performance reconfigurable computing systems can be used in the ground to address the Earth system modeling challenges.

The challenges to harness the potential of reconfigurable computing to benefit NASA's technology goals, however, are yet to be achieved. Reconfigurable computing was enabled by the commercial implementation of FPGA devices beginning in 1985. Academic, government and commercial research entities have explored extensively the development and utilization of this technology. During the 1996 fiscal year, the Adaptive Scientific Data Processing (ASDP) project at Goddard Space Flight Center began the investigation of the use of reconfigurable computing technology for satellite data processing[6]. The project lasted 3 years and successfully identified the potential benefits of the technology. The ASDP created a knowledge base and prepared the ground for further action. The ASDP worked in cooperation with the Adaptive Computing System (ACS) program of the Defense Advance Research Projects Agency (DARPA). The DARPA ACS project invested close to a hundred million dollars in the advancement of reconfigurable computing [7]. But despite all these efforts, reconfigurable computing is still not ready for infusion in NASA's missions. Further development of the technology is still required.

In order to better understand and harness the potential of reconfigurable computing, it is necessary to determine where reconfigurable computing stands in comparison to other modes of computing. Most everyone in the reconfigurable computing field recognizes that reconfigurable computing enables a new computing paradigm [22, 23, 24, 32]. There are two prevalent computing paradigms in development: the hardware and the software paradigms. Reconfigurable computing technology enables a third one, which we name the configware paradigm.

The resources NASA reserves to address the development of reconfigurable computing technology can be greatly enhanced by the recognition that a new computing paradigm is in evolution. Thus, the task to develop a roadmap for the development of reconfigurable computing technology must look at the evidence that points towards the existence of the configware paradigm. It is also necessary to identify and evaluate the current state of the technology, the commercially available products, the development trends, and the players involved in the research and development of the technology.

The Goddard Space Flight Center (GSFC) technology mission is to provide Agency leadership to advance next-generation spacecraft, sensor, and instrument technology [3]. GSFC leadership in the investigation of reconfigurable computing was established with the ASDP project. The Center has a worldwide-recognized knowledge base and an established reputation within the reconfigurable computing community. Any action to further the development of reconfigurable computing and its application to spacecraft data processing will benefit from this reputation and knowledge.

This report presents an overview of the evolution of new paradigms and looks at the evidence that shows the evolution of the configware paradigm. It briefly presents an overview of the current state of reconfigurable computing technology and provides the references to more detailed information. The application of reconfigurable computing in space is analyzed next. Finally, some propositions are suggested for NASA organizations to further develop the technology in order to enable its infusion in NASA's missions.

The Evolution of Paradigms

The evolution of a system is said to pass through three distinct phases [11, 12]. In phase 1, the formation phase, the original standard is invented or formed from the combination of several elements previously disorganized. In phase 2, the consolidation phase, the new standard is perfected in a cumulative process through repetitive modifications and rejection of the pieces that do not fit the original standard. In phase 3, the transformation phase, the original standard is broken, and differences previously left out are incorporated through innovation and opening of the system. Invention, perfection, and innovation characterize the three phases of development of a system.

The three phases of the evolution of a system can be illustrated by what is called an S curve, as shown in figure 1. The phases are marked by two breakpoints, or points of change.

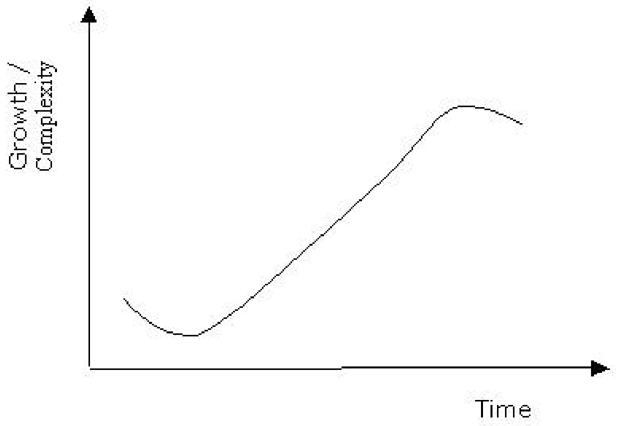

Figure 1 – S Curve of a System in Evolution

The first breakpoint marks the change from phase 1, where an accepted paradigm has been established, and phase 2, where the goal is to improve the established paradigm. The first phase is characterized by the divergence of ideas while the second phase is marked by convergence. It is only possible to reach higher levels of growth through repetition, amplification and perfection of the basic paradigm. However, every successful system enjoying the growth in phase 2 extends its reach to a certain level where the original paradigm is exhausted, and it needs to be changed. At this second breakpoint, the ideas that have been rejected in phase 2 are revisited.

Two distinct processes characterize phase 3, the transformation phase. The differences are integrated through the means of innovation and construction following the basic paradigm that appeared in phase 2, and the old system is continuously integrated through assimilation and connection of the parts previously left out. At the same time, a new phase 1 is created, and a bifurcation appears, as illustrated in figure 2. There is a division, where part of the established system moves away from the process of innovation and begins a process of reinvention of the entire system in a much more complex level.

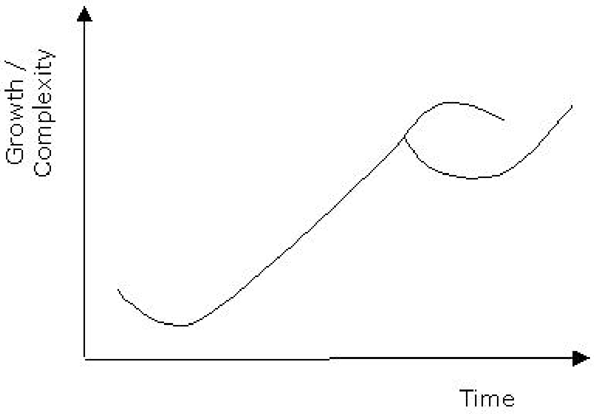

Figure 2 – A Paradigm Shift

However, the new emergent cycle, or the new phase 1, is invisible, covered by the current and successful paradigm, and clouded by its own state of disorder. As it reaches its phase 2, it suddenly appears from nowhere. It is very common that organizations and entire industries fail to benefit from the bifurcation process, and participate in the creation of a new phase 1. History shows that the owners of railroads do not create airlines, and the manufacturers of mechanic calculators do not launch electronic computers. Only the people outside of these organizations and industries are sufficiently free from the past to recreate the future.

Reconfigurable Computing and the Configware Paradigm

The Von Neumann compute engine, as seen in figure 3, enabled the creation of the software paradigm, where an algorithm is mapped into code that is interpreted by a processor. Microprocessors such as the Intel Pentium are a representation of the Von Neumann compute engine. In the hardware paradigm, an algorithm is mapped into storage and logic functions that are isomorphic with the computation. The Application Specific Integrated Circuits (ASIC) are designed using the hardware paradigm [15, 16, 33].

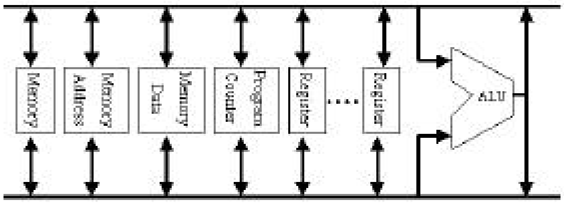

Figure 3 – The Von Neumann Engine Datapath

Reconfigurable computing utilizes the hardware paradigm to build applications. However, a reconfigurable FPGA allows a new configuration to define its logic circuit, or *hardware* design, in a fraction of time. As a result, a reconfigurable computing engine allows multiple *hardware* designs to be time multiplexed and enables several applications to share the same device, just like a microprocessor. Suddenly, the Von Neumann model disappears from the picture. The software paradigm as defined by the Von Neumann model no longer applies and a new and more complex one is born, the configware paradigm.

The Von Neumann compute engine allows very large applications to be built using the same processor. Algorithms are broken into a series of instructions that manipulate data stored in memory elements. The larger the application is the longer is the sequence of instructions, which results in a longer time to execute and deliver a result. In a configware engine, the application is divided in blocks that can fit into a single FPGA device. The blocks are time-multiplexed, while the intermediary results are stored in memory. The larger the logic capacity of the FPGA is, the larger is the block of the application that can be executed at a time, and the shorter is the time to execute the entire application. Also, the faster an FPGA can switch between configurations, the faster the entire application will execute and deliver a result. The standard model that will define the configware paradigm is yet to be invented. Several elements have been constructed but the integration of these elements into a single architecture that can support continuing growth has not yet been done.

The Computing Paradigm Shift

The mathematician John Von Neumann developed his computing model in the late 1940s [14, 17]. Since then, the field of computer science and engineering experienced an enormous growth and it has provoked deep changes in the way of life of peoples and societies around the world. Computers and the networks that connect them are about to integrate the entire planet in what is sometimes called the Earth's nervous system. Is the software paradigm in expansion or is it reaching exhaustion? No one knows for sure. Most studies indicate that it is in clear progression [9, 10]. These studies, however, look at past performance and current trends. Alternative paths are rarely investigated.

The engineer Gordon Moore made an observation in 1965 that the number of transistors per integrated circuit would double every 18 months [13]. He predicted that the trend would continue through 1975. However, what is called Moore's Law has persisted to this date, and the microprocessors that enable the software paradigm have benefited from this trend.

Nevertheless, the growth in the number of transistors per integrated circuit also benefits the development of FPGAs, the devices that enable reconfigurable computing. The next question then becomes: Is reconfigurable computing going to be integrated as part of the current computing paradigm and support its growth? That is very possible and a lot of the research taking place in the reconfigurable computing field is to integrate reconfigurable logic and microprocessors in the same silicon die [25, 29]. There are also attempts to use the current application development tools and target solutions to FPGAs [26, 27, 28]. For example, there are a considerable number of attempts to map C language descriptions to FPGA logic. The results so far have been disappointing.

The other possibility is that reconfigurable computing be abandoned as a practical solution at this time since it does not contribute to the growth of the current computing paradigm. In a later time, when the Von Neumann paradigm shows exhaustion, reconfigurable computing would be revisited as an option to either strength or replace the Von Neumann paradigm. Given the investment and the reach so far achieved by the field of reconfigurable computing, this is not likely to happen.

Reconfigurable computing has experienced a tremendous development in the past 5 years. Several technology research and development organizations have introduced reconfigurable computing in their research agenda [7]. The breadth and depth of topics within the field of reconfigurable computing are becoming too large and complex for a single individual to master them all. These are the characteristics of a new paradigm in evolution, currently in the phase of invention, or phase 1. The divergence of ideas and concepts are prevalent in the reconfigurable computing community. But despite all these efforts, a new standard is yet to be established to push ahead the configware paradigm.

The State of Reconfigurable Computing

The logic density of an FPGA device is a great determinant of its capabilities. The logic gate is the unit of measurement of the logic density of an FPGA. It is desired that the device logic density allow a considerable portion of an application to be implemented so that partition of an application among several FPGAs is not needed. The first application developed by the ASDP group in 1996 used a 13,000-gate device, the largest device available then. Today, the logic density of commercially available FPGA devices is above 3-million logic gates. The trend is for the gate density to double every 18 months, following Moore's Law. This trend persisting, in 2010 FPGAs will present a density of 192-million gates as illustrated in the following figure.

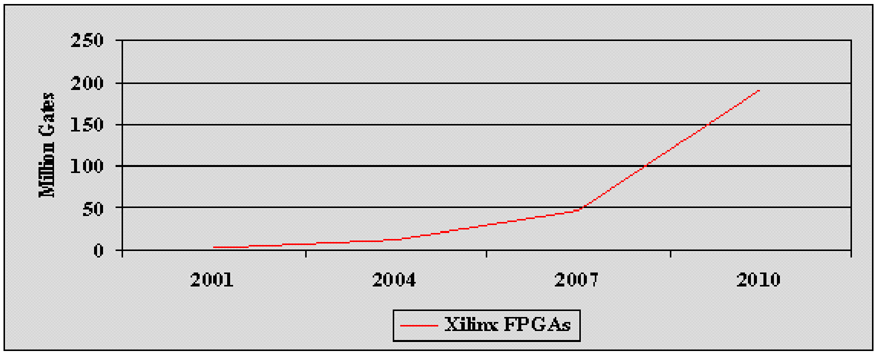

Figure 4 – Projection of the Gate Density Growth of Xilinx FPGAs

Systems Architectures

The FPGA logic density has reached a level that allows to conceive a reconfigurable computing system composed of a single FPGA that is capable of running a considerable number of applications. As the logic density of the FPGA devices increases, larger systems will fit in a single device. The configuration of the FPGA can be stored in local non-volatile memory or passed to it via a communication port such as a modem or Internet connection. One can envision a reconfigurable system where at boot time the system is configured to ask the user which application he/she wants to run, and upon command, build a TCP/IP port that connects to an application server, downloads the application requested by the user, and configures itself to run the application and interact with the user.

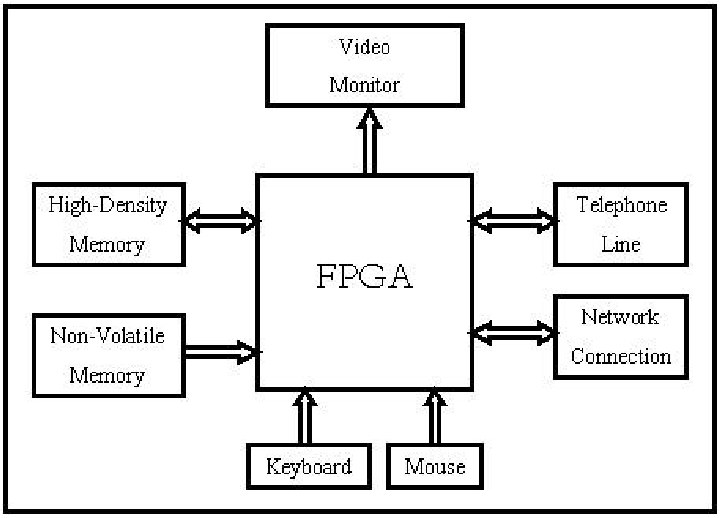

Figure 5 – Single-FPGA Reconfigurable Computer

The single-FPGA architecture, however, has become feasible only recently, approximately one year. Before then, the systems solutions encompassed a combination of microprocessors and FPGAs. The simplest solution is to connect a single FPGA to the microprocessor input/output bus. A variation would be to use a digital signal processor (DSP) instead of a general-purpose microprocessor. There are several research and commercially available boards which use all types of microprocessors, DSPs, and FPGAs. There are even certain architectural combinations that include all three types of processing elements [25, 26, 27, 29, 30, 31, 32].

If one wants to design a reconfigurable computing application that is larger than the single FPGA device, the solution is to use a system architecture that has more than one FPGA. Once a system has more than one FPGA, the interconnection topology of the FPGA devices becomes an issue of concern. There are basically two types of interconnection schemes: the bus oriented architecture where one or more buses are used to interconnect the various FPGAs in the system (shared or systolic bus); and the switch matrix architecture where the devices can be interconnected according to the configuration of the switch matrix.

Due to architectural limitations of current FPGA devices, it is always required to have high-density external memories attached to the FPGAs. These memory devices can be either static RAM for fast access or dynamic RAM to allow higher density and lower cost memory blocks. This further complicates the system architecture since the application has to be properly mapped to the FPGAs and memory devices.

The multi-FPGA systems are normally designed to operate as co-processors in a computer system. These co-processors are attached to the computer input/output bus and the applications are partitioned between the host computer and the co-processor. For a very large application, it might be necessary to attach several co-processors to the host computer bus. It is also possible to have a multi-computer system connected by a network and operating as a parallel processing system. In such a system, each computer hosts one or more FPGA co-processors.

The FPGA co-processors also have input/output ports to integrate data acquisition platforms directly into the system. This solution allows for a higher rate of data transfer and eliminates the need to go through the host computer bus, which normally is a source of bottlenecks. The host computer has a limited bandwidth and for applications that require high amounts of data to be transferred back and forth to the FPGA co-processor, the host bus bandwidth can represent a limitation on the utilization of the FPGA co-processor.

Development Tools

In either a multi-FPGA or a single-FPGA architecture, if the system is to reside on a host computer and operate as a co-processor, then the vendor is required to provide device drivers to communicate between the host and the FPGA co-processor. They also supply a library of functions in a high-level language to facilitate the host application development. If the FPGA co-processor is to be able to operate in several computer architectures using different operating systems, then the vendor must supply device drivers and function libraries for all architectures and operating systems it wishes to support.

Traditionally, the FPGA application design is implemented using hardware design tools. These tools include FPGA place and route, schematics diagrams, functional and timing simulators, and hardware description languages. The host application is normally designed utilizing a high level language such as C, C++, or Java. There are new tools under development and some already being commercialized that are a mix of these traditional hardware and software tools [22, 23, 28].

The two dominant Hardware Description Languages (HDLs) are VHDL and Verilog. In an HDL a parallel model is used to specify digital circuits. These languages operate at the gate level and can be seen as a textual version of a schematic editor. They are used for board-level, integrated circuit and FPGA design.

High Level Languages (HLLs) are standard programming languages invented for microprocessor-oriented architectures, i.e, architectures that have a predefined instruction set. HLLs such as C, Java, FORTRAN, and Cobol, provide a sequential model for specifying algorithms. This is due to the fact that microprocessors are inherently sequential machines that despite a high-degree of internal functional parallelism, they divide an application into a sequence of steps. In comparison to HDLs that work at the gate level, HLLs work at the instruction level.

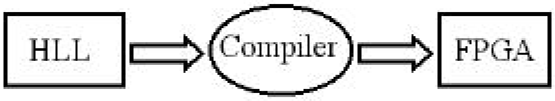

Figure 6 – HLL to FPGA Compiler

One approach to designing for FPGAs is to devise a compiler that translates HLL specifications directly to FPGAs (see figure 6). This method basically substitutes the HDL for the HLL. The compiler generates netlists instead of instructions. To create such a compiler requires a syntax transformation on the original HLL to accommodate the HDL characteristics. Even though this solution provides a more algorithm oriented emphasis, instead of the circuit design, the transformed HLL becomes more like the HDL with less capability than the standard HDLs and the original HLL. A good HLL-to-FPGA compiler is nothing more than a new HDL with the syntax close to the original HLL. It is useful to attract the attention of software engineers used to the syntax of HLLs, but it hardly convinces hardware engineers of the need of learning a new HDL that most probably is weaker than the standard HDLs available. Examples of hardware design HLLs are: PamDC, Transmogrifier C, Ruby, Lava, Pebble, Lola, PLASMA, spC, rALU, and others.

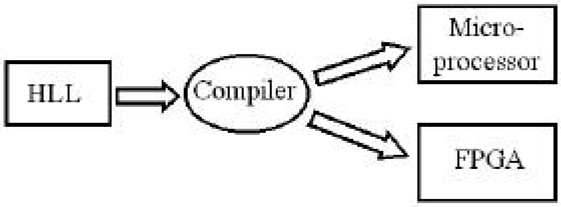

Figure 7 – HLL to Microprocessor and FPGA Compiler

There are also HLLs targeted at system level designs that combine both microprocessors and FPGAs (figure 7). The goal in such languages is to have an application specification environment that allows the automatic partitioning of the application between the microprocessor and the FPGA. These languages can be used both for FPGAs and ASICs to address the concept of System on Chip (SOC).

Some early system level tools are Prism (C compiler targeted to Motorola 68K/Xilinx 4K system) and Data Parallel C (SIMD C targeted to Splash 2). There are currently several HLL tools, some experimental others commercially available. They are Hendel-C, JHDL, SpecC, SystemC, C2VHDL/C2Verilog, Superlog, CynLib, Forge, and FilerExpress.

An application design approach that is rather compelling for the scientist or algorithm designer is one that utilizes mathematical or data-flow tools and languages to specify the application and target it directly into FPGAs. Three of these design tools being developed by researchers to create reconfigurable computing tools include: Matlab, Khorus and Ptolomey. Matlab is a text-based mathematical language widely used by scientists and engineers for modeling and algorithm description. Khorus is a dataflow diagram tool for image processing. Ptolomey is a graphical entry tool for systems level design.

Reconfigurable Computing in Space

Onboard processing of sensor data through the use of high-performance computing architectures is among the requirements for the generation of data products for direct distribution to users. Other applications of onboard processing are the autonomous navigation of satellite clusters, intelligent sensor control, and the processing of hyperspectral imagery and radar data.

Increasing the computing capability of the spacecraft, however, requires the creation of space-qualified versions of the devices used in the ground. The most advanced radiation hard microprocessors available today deliver 20-30 MIPS (RH-32 and RAD6000) and 50 MFLOPS (RH C40) [8]. New versions to be available around 2001/2002 will deliver 200-300 MIPS (Power PCs 603 and 750, and Pentium) and 1-2 GFLOPS (TigerSHARC DSP). Future programs project the availability of space-qualified processors that will deliver 3-4 GIPS before the year 2010.

However, the time lag between the availability of a microprocessor for ground applications and the same microprocessor being available for space is about 6 to 8 years at best. As a result while the performance of a microprocessor for ground applications doubles about every 18 months, it takes much longer for that performance increase to reach the spacecraft. In addition, the performance of a single microprocessor is not always enough to address the requirements of the applications in question.

It is well known that high-performance computer architectures are solely driven by the applications in question. There is not a generalized solution for all applications. The application developer should first understand the algorithm and then choose the solution among the implementation options available. A small set of options reduces the capability of the designer to devise the best solution.

High-performance computing has been addressed in several forms: through the design of Application Specific Integrated Circuits (ASICs) for well defined algorithms such as data compression and decompression, through the use of specialized Digital Signal Processors (DSPs), and through the association of several microprocessors forming a parallel computer. In the space environment, however, there are not as many options since there are not as many space-qualified devices for use. As a result, spacecraft onboard processing is limited not only by the current availability of the fastest devices, but also by the number of options to compose a high-performance computer architecture.

Reconfigurable computing technology has the potential to bring high-performance computing to the spacecraft at a reduced cost. Many examples of reconfigurable computing applications have shown an increase of several orders of magnitude performance over microprocessor-based solutions. The ASDP project has investigated the use of reconfigurable computing in spacecraft data processing. The work concentrated in the acceleration of the initial stages of the processing pipeline (Level 1 and 2) [19, 20, 21]. It was shown that reconfigurable computing presents performance increases over an order of magnitude when compared to microprocessor-based solutions.

Device reliability is the major concern for the application of reconfigurable computing in harsh environments such as space. The goal of reliability studies undertaken by reconfigurable computing researchers is to produce self-healing devices and systems that take advantage of the ability of FPGAs to be in-circuit reconfigurable. As faults are detected, circuit paths are adjusted to work around the defects. Several techniques have been developed, but there are still issues to be resolved to enable a complete self-healing architecture. It is in this area that FPGAs may offer a great advantage over other silicon devices for space use. The use of the FPGA's intrinsic self-healing characteristics leads to a shorter cycle to enable their usage in space if the process of traditional radiation hardening is not required. The result could be the establishment of a new paradigm for computing power onboard the spacecraft. Reconfigurable computing, if enabled for spacecraft usage, can not only offset the difference between the computing capabilities of the ground systems and the spacecraft, but can also set a pace for development of spacecraft computing capabilities similar to that found for ground systems today.

Technology Roadmap

The realization that reconfigurable computing enables a new computing paradigm should be the determinant factor in the establishment of a roadmap for the development and infusion of the technology in NASA's programs. NASA can not only benefit from the potential of reconfigurable computing for the improvement of its scientific missions, but it can also make fundamental contributions to the development of a new computing paradigm that will greatly influence human and technological progress.

The fact that a new paradigm is in evolution demonstrates the need for a new mindset to think new concepts that break away from the current paradigm. An engineering organization that is itself structured to reflect the hardware and software functions is going to resist the realization of projects that question its infrastructure. It is necessary to create an independent project that is not influenced by the pressures to keep the status quo.

It is also important to recognize that a new paradigm is always more complex than the current one. As such, the more diverse is the group of people involved in the task of determining the new paradigm, the larger is the chance for success in a shorter period of time. Bringing together scientists of all fields, mathematicians, software engineers, hardware engineers, and systems engineers is very important. The birth of the software paradigm was in great part a result of the Macy Conferences that took place between 1946 and 1953, which brought together scientists of all fields and started the field of cybernetics. The availability of the Internet for online discussions and cooperative development and distribution of knowledge makes the task of creating a new paradigm easier to accomplish in our days. A project sponsored by NASA would have the credibility to bring all these elements together.

Evaluating and integrating the diversity of actions that occur in the reconfigurable computing field is important for the establishment of a measurement of progress. NASA should have an ongoing test-bed to benchmark FPGA device architectures, reconfigurable systems architectures, development tools, device reliability techniques, and other issues that come to happen in the field. Besides testing and evaluating the accomplishments of third party developers, NASA should also utilize the current state of the technology to implement some of its applications. The experience of the ASDP project in using reconfigurable computing to develop space applications were greatly appreciated by researchers of the DARPA ACS project while developing new tools for application design. It is important to continue this interaction and organize the information in a knowledge database made available over the Internet (see http://fpga.gsfc.nasa.gov).

In short, NASA needs to reinstate the actions of the ASDP project and broaden the activities that were in place while the project was alive. Integrating the efforts within NASA and the research community outside NASA, and expanding the development efforts to other areas of science and engineering is essential for a successful technology roadmap. These activities can be greatly facilitated through the establishment of an open source project to create tools and components to form a public library of reconfigurable computing functions. There are several examples of open source projects that enabled the development of reliable software in a short period of time, for a cost close to zero [18]. The Linux operating system is the most popular one. An open source project could represent the savings of many years in the establishment of the configware paradigm. NASA should sponsor such an endeavor.

The NASA New Millennium Program (NMP) conducts space flight validation of breakthrough emerging technologies that will significantly benefit future space science and Earth science missions [35]. The NMP selects for test high-payoff but risky technologies, those that may present a risk to any mission using it for the first time in space. Reconfigurable computing is one of the technologies that represent a breakthrough in the scientific capability of spacecrafts. The NMP program should undertake a mission that utilizes the full capabilities of reconfigurable FPGAs. It would represent a perfect opportunity to bring together scientists and engineers to explore the potential of the new technology in a new mindset. The effort could result in the development of the configware paradigm.

A concept in evidence today is the single-chip spacecraft based on a high-density FPGA (multi-million gate device). Figure 8 illustrates this concept. Similar to the single-FPGA computer concept, the single-FPGA spacecraft breaks many of the paradigms in spacecraft design. First of all, all devices are connected directly to the FPGA and any protocol adjustment is done within the FPGA. No interface chips between the instruments and the FPGA are required. It is a bus-less architecture, totally reconfigurable and adaptive. It does not require an operating system, and all communication protocols are possible since they are implemented in a reconfigurable FPGA. A self-checking and self-healing architecture is possible and reduces the need for radiation hardening of the parts. A new mission sponsored by the NMP program should investigate this architecture.

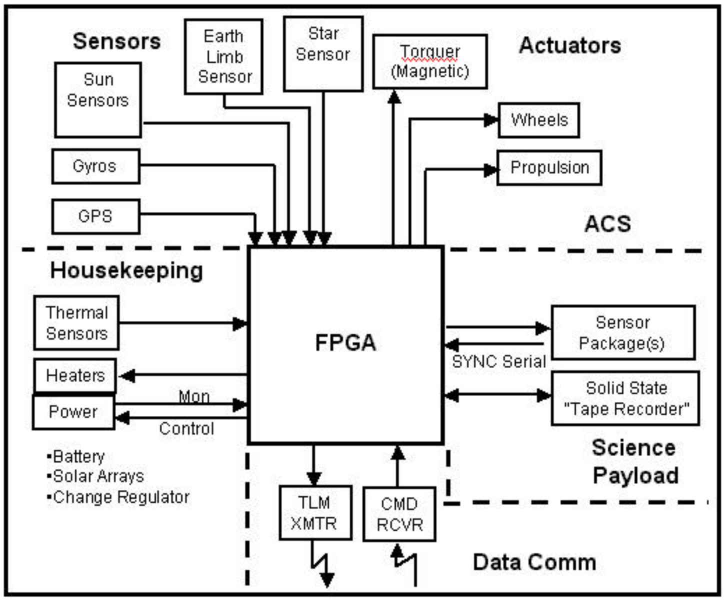

Figure 8 – Single-FPGA Spacecraft

The single-FPGA computer and the single-FPGA spacecraft concepts open the way to a totally new engineering of the ground and spacecraft systems. Given the added level of flexibility and computing power offered by FPGAs, it is possible to envision a more integrated space-ground system, where both parts present similar computing capabilities. A scientific mission that explores new mathematical concepts and is not pressured to be built in the current computing paradigm is the preferred partner for this technology development endeavor.

Conclusion

There is enough evidence to indicate that the advent of reconfigurable computing is creating the configware paradigm. Breaking away from the current mindset requires more than a traditional technology development and infusion approach. It requires a managerial commitment to a long-term plan to explore new thinking in computer science and engineering.

The on-going development of the reconfigurable computing field is full of excitement, divergence of ideas and approaches, and much expectation that the new paradigm is about to emerge. Nobody knows, however, how long it will take for this to happen. A short-term approach to research and development could only benefit those ideas that are intended to promote the progress of the current paradigm. Much of the research done to date follows this track of short-term compensation. A responsible, long-term approach, realizes that it could take more than a decade for the new paradigm to emerge. This understanding would best benefit the actions that can truly contribute to the creation of the configware paradigm.

The responsibility to create a new computing paradigm might be too big a task for NASA. But the potential benefits that the new paradigm brings would enable NASA to address scientific challenges not possible or envisioned today. NASA can sponsor, support, or steer a long-term, methodic approach to characterize, document, and facilitate the evolution of the configware paradigm. Ultimately, the methodology adopted will reflect NASA's level of commitment to meet the challenges to develop and enhance human scientific and technological knowledge.

References

1. NASA Strategic Plan 2000 - http://www.hq.nasa.gov/office/codez/plans/pl2000pdf
2. NASA Earth Science Enterprise Strategic Plan 2000 - http://www.hq.nasa.gov/office/codez/plans/ESE00plan.pdf
3. Goddard Space Flight Center – Implementing NASA's Strategies for the 21st Century - http://www.hq.nasa.gov/office/codez/plans/GSFCImp98.pdf
4. NASA Technology Plan - http://technologyplan.nasa.gov/
5. Figueiredo, M.; Introduction to Reconfigurable Computing; Paradigm Computing Corporation, June 2001
6. Figueiredo, M.; The NASA Adaptive Scientific Data Processing (ASDP) Project; Paradigm Computing Corporation, April 2001
7. Figueiredo, M.; The DARPA Adaptive Computing Systems Program; Paradigm Computing Corporation, August 2001
8. The Aerospace Corporation; Advanced Information Systems Technology Projections; November 2000
9. Simon, D. H.; The Future of Scientific Computing; Lawrence Berkeley National Laboratory, June 6, 2000. http://www.nersc.gov/aboutnersc/presentations/future/index.htm
10. Air Force 2025, Federation of American Scientists, http://www.fas.org/spp/military/docops/usaf/2025/

11. Henderson, Hazel; Paradigms in Progress; Berrett-Koehler, San Francisco, 1995
12. Land, George; Jarman, Beth; Breakpoint and Beyond: Understanding and Shaping the Forces of Change; 1990
13. Moore, G.; Moore's Law; http://www.intel.com/research/silicon/mooreslaw.htm
14. Heims, Steve J.; John von Neumann and Norbert Wiener; MIT Press, Cambridge Mass. 1980
15. Tucker, A. B.; The Computer Science and Engineering Handbook; CRC Press, 1996
16. Oldfield, J. V., Dorf, R. C.; Field Programmable Gate Arrays: Reconfigurable Logic for Rapid Prototyping and Implementation of Digital Systems; John Wiley and Sons, 1995.
17. Von Neumann, J; The Computer and the Brain; Yale University Press, New Haven, 1958.
18. Sandred, J.; Managing Open Source Projects; John Wiley and Sons, 2001, ISBN 0-471-40396-2
19. Figueiredo M. A., Stakem P. H.; Extending NASA's Data Processing to Spacecraft, IEEE Computer Volume 32 number 6, pages 115-118, June 1999.
20. Figueiredo M.A.; Winiecki K.B.; Graessle T.L.; Analysis of the Applicability of Reconfigurable Computers in Satellite Telemetry Data Processing; Proceedings of SPIE Vol 3526 - Configurable Computing: Technology and Applications, Boston, Massachusetts.
21. Figueiredo M. A.; Implementation of a Probabilistic Neural Network for Multi-Spectral Image Classification on a FPGA Custom Computing Machine, 5th Brazilian Symposium on Neural Networks - Belo Horizonte, Brazil, December 9-11, 1998.
22. Villasenor, J., Hutchings B.; The Flexibility of Configurable Computing, IEEE Signal Processing Magazine, pages 67-84, September 1998.
23. Tredennick, N.; Get Ready for Reconfigurable Computing; Computer Design, pages 55-63, April 1998.
24. Hauck, S.; The Future of Reconfigurable Systems; Keynote Address, 5th Canadian Conference on Field Programmable Devices, Montreal, June 1998
25. Hauck, S; The Roles of FPGAs in Reprogrammable Systems; Proceedings of the IEEE, Vol. 86, No 4, pp 615-639, April, 1998, http://www.ee.washington.edu/faculty/hauck/publications/mFPGAhard.pdf

26. Compton, K.; Hauck, S.; Reconfigurable Computing: A Survey of Systems and Software; http://www.ee.washington.edu/faculty/hauck/ConfigCompute.pdf
27. S. Hauck, A. Agarwal, Software Technologies for Reconfigurable Systems, Northwestern University, Dept. of ECE Technical Report, 1996. http://www.ee.washington.edu/faculty/hauck/publications/mFPGAsoft.pdf
28. Guccione, S.; Run Time Reconfiguration at Xilinx; Reconfigurable Architectures Workshop 2000
29. Vueillemin, J. E.; Bertin, P.; Roncin, D.; Shand, M.; Touati, H. H.; Boucard, P.; Programmable Active Memories: Reconfigurable Systems Come of Age; IEEE Transactions on VLSI Systems, Vol. 4, No 1, March 1996
30. Van Den Bout, D.; Morris, J. N.; Thomae, D.; Labrozzi, S.; Wingo, S.; Hallman, D.; AnyBoard: An FPGA-Based, Reconfigurable System; IEEE Design and Test of Computers, pages 21-30, September 1992
31. Villasenor, J; Schoner, B.; Chia, K; Zapata C.; Kim, H. J.; Jones, C.; Lansing, S.; Mangione-Smith; Configurable Computing Solutions for Automatic Target Recognition; IEEE Symposium on FPGAs for Custom Computing Machines, pp 70-79, 1996
32. Buell, D. A.; Arnold, J. M.; Kleinfelder, W. J.; Splash 2: FPGAs in a Custom Computing Machine; IEEE Computer Society Press, Los Alamitos, California, 1996
33. Trimberger, S. M. ed.; Field Programmable Gate Array Technology; Kluwer Academic Publishers, Boston, 1994
34. Guccione, S.; Programming Fine-Grained Reconfigurable Architectures – PhD Thesis; http://www.io.com/~guccione/Diss/Diss.html
35. The New Millennium Program; http://nmp.jpl.nasa.gov/